Asterism: an integrated, complete, and open-source approach for running seismologist continuous data-intensive analysis on heterogeneous systems

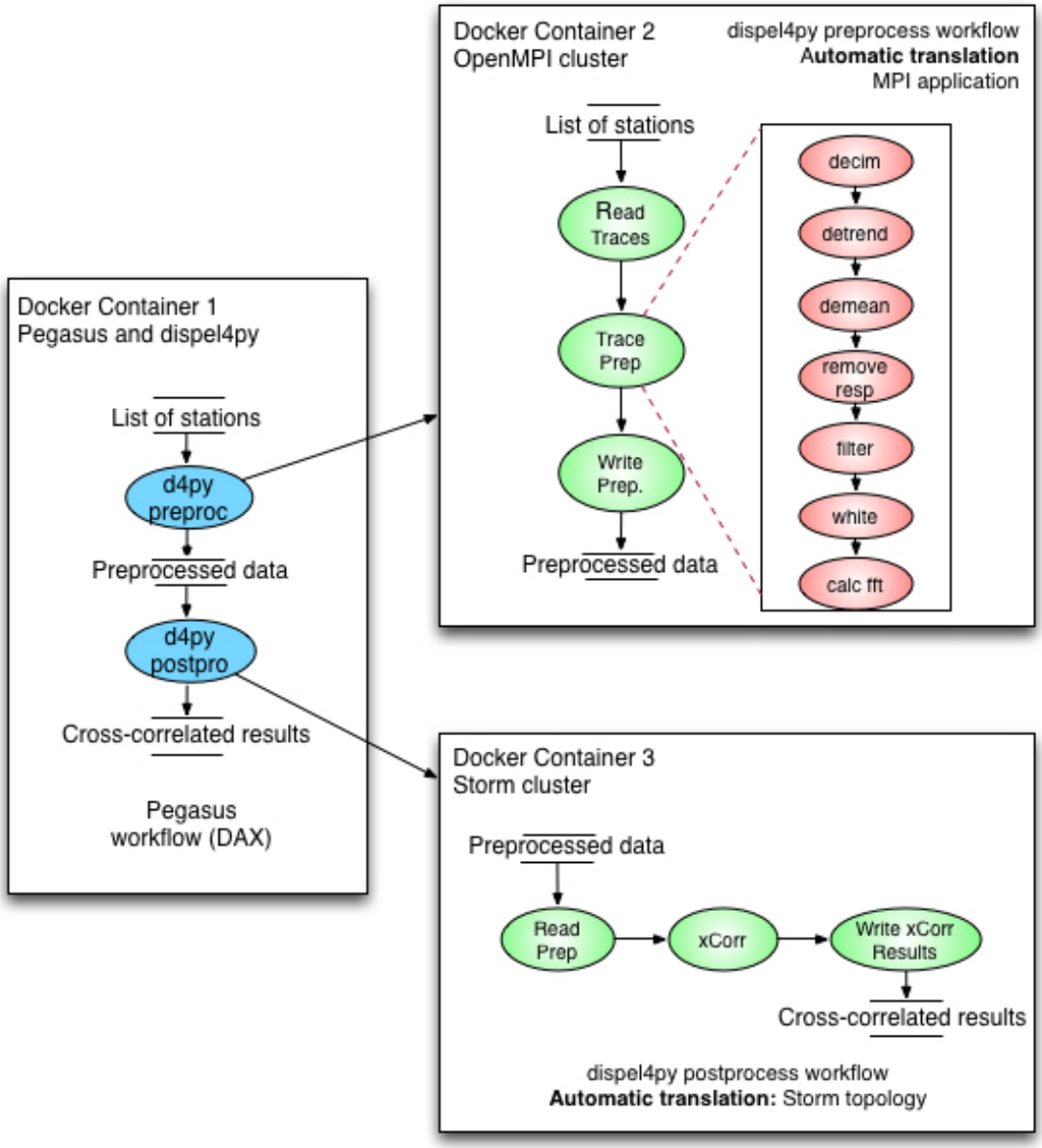
Asterism: an integrated, complete, and open-source approach for running seismologist continuous data-intensive analysis on heterogeneous systems

We present *Asterism*, an open source data-intensive framework, which combines the **Pegasus** and **dispel4py** workflow systems. *Asterism* aims to simplify the effort required to develop data-intensive applications that run across multiple heterogeneous resources, without users having to: re-formulate their methods according to different enactment systems; manage the data distribution across systems; parallelize their methods; co-place and schedule their methods with computing resources; and store and transfer large/small volumes of data.

Asterism's key element is to leverage the strengths of each workflow system: *dispel4py* allows developing scientific applications locally and then automatically parallelize and scale them on a wide range of HPC infrastructures with no changes to the application's code; *Pegasus* orchestrates the distributed execution of applications while providing portability, automated data management, recovery, debugging, and monitoring, without users needing to worry about the particulars of the target execution systems. *Asterism* leverages the level of abstractions provided by each workflow system to describe hybrid workflows where no information about the underlying infrastructure is required beforehand.

The feasibility of *Asterism* has been evaluated using the *seismic ambient noise cross-correlation* application, a common data-intensive analysis pattern used by many seismologists. The application preprocesses (*Phase1*) and cross-correlates (*Phase2*) traces from several seismic stations. The *Asterism* workflow is implemented as a *Pegasus* workflow composed of two tasks (*Phase1* and *Phase2*), where each phase represents a *dispel4py* workflow. *Pegasus* tasks describe the in/output data at a logical level, the data dependency between tasks, and the e-Infrastructures and the execution engine to run each *dispel4py* workflow.

We have instantiated the workflow using data from 1000 stations from the IRIS services, and run it across two heterogeneous resources described as Docker containers: MPI (*Container2*) and Storm (*Container3*) clusters (**Figure 1**). Each *dispel4py* workflow is mapped to a particular execution engine, and data transfers between resources are automatically handled by *Pegasus*. *Asterism* is freely available online at http://github.com/dispel4py/pegasus_dispel4py.


Docker Container 1
Pegasus and dispel4py
List of stations
d4py preproc
Preprocessed data
d4py postpro
Cross-correlated results
Pegasus workflow (DAX)
Docker Container 2
OpenMPI cluster
dispel4py preprocess workflow
Automatic translation
MPI application
List of stations
Read Traces
Trace Prep
Write Prep.
Preprocessed data
decim
detrend
demean
remove resp
filter
white
calc fft
Docker Container 3
Storm cluster
Preprocessed data
Read Prep
xCorr
Write xCorr Results
Cross-correlated results
dispel4py postprocess workflow
Automatic translation: Storm topology